Listing of claims:

1. (Currently amended) A method for representing header and footer structures in a markup language document, comprising:

determining properties corresponding to a mini-document that relates to at least one section of an application document, wherein the mini-document includes a body portion, wherein the mini-document includes at least one member of a group comprising: a header and a footer;

mapping the properties of the mini-document into a markup language element, wherein mapping the properties includes mapping a type attribute that corresponds to an occurrence pattern of the body of the mini-document within the application document, wherein the type attribute causes the body portion to be repeated in the application document in accordance with the occurrence pattern indicated by the type attribute ~~wherein mapping includes mapping the properties into a context free chunk element~~; and

storing the properties of the mini-document in the markup language document.

2-3. (Cancelled)

4. (Currently amended) The method of Claim 1, wherein the type attribute corresponds to whether the mini-document occurs on at least one member of a group comprising: ~~a first page~~, odd pages of the specified section of the application document, or even pages of the specified section of the application document.

5. (Cancelled)

6. (Previously presented) The method of Claim 1, further comprising:

determining properties corresponding to an additional mini-document that relates to at least one section of the application document;

mapping the properties of the additional mini-document into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and storing the properties of the additional mini-document in the markup language document.

7. (Original) The method of Claim 1, further comprising:

determining whether properties associated with all mini-documents of the application document have been stored in the markup language document; and

processing further mini-documents when the properties associated with all mini-documents have not been stored in the markup language document.

8. (Original) The method of Claim 1, wherein the properties of the mini-document stored in the markup language document are understood by an application that understands the markup language when the mini-document is not native to the application.

9. (Original) The method of Claim 1, wherein the markup language document is manipulated on a server to substantially reproduce the mini-document of the application document notwithstanding the presence of an application that generated the markup language document.

10. (Currently amended) A computer-readable medium for representing headers and footers in a markup language document, comprising:

determining properties relating to a mini-document used within a word-processing document, wherein the mini-document includes a body portion having text;

determining whether the mini-document is at least one member of a group comprising: a header and a footer;

writing the properties into a markup language element, wherein writing the properties includes mapping a type attribute that corresponds to an occurrence pattern of the text of the body portion of the mini-document within the word-processing document, wherein the type attribute causes the same text of the body portion to be repeated in the application document in accordance with the occurrence pattern indicated by the type attribute~~ wherein writing includes writing the properties into a context free chunk element~~; and storing the properties in the markup language document such that the headers and footers of the word-processing document are substantially maintained when the markup language document is parsed by an application.

11. (Original) The computer-readable medium of Claim 10, wherein the markup language document is manipulated on a server to substantially reproduce the mini-document of the word-processing document notwithstanding the presence of an application that generated the markup language document.

12. (Original) The computer-readable medium of Claim 10, wherein the properties of the mini-document stored in the markup language document are understood by an application that understands the markup language when the mini-document is not native to the application.

13. (Cancelled)

14. (Currently amended) The computer-readable medium of Claim [[13]] 10, wherein the type attribute corresponds to whether the mini-document occurs on at least one member of a group comprising: ~~a first page~~, odd pages of the specified section of the application document, or even pages of the specified section of the application document.

15. (Cancelled)

16. (Previously presented) The computer-readable medium of Claim 10, further comprising:

determining properties corresponding to an additional mini-document that relates to at least one section of the word-processing document;

mapping the properties of the additional mini-document into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and storing the properties of the additional mini-document in the markup language document.

17. (Original) The computer-readable medium of Claim 10, further comprising:

determining whether properties associated with all mini-documents of the word-processing document have been stored in the markup language document; and

processing further mini-documents when the properties associated with all mini-documents have not been stored in the markup language document.

18. (Currently amended) A system for representing header and footer information in a markup language document, comprising:

a processor; and

a memory associated with computer-executable instructions configured to:

determine properties relating to a mini-document included in at least one section of an application document, wherein the mini-document includes a body portion having text;

determine whether the mini-document is at least one member of a group comprising: a header and a footer;

map the properties into a markup language element, wherein mapping the properties includes mapping a type attribute that corresponds to an occurrence pattern of the text of the body portion of the mini-document within the application document, wherein the type attribute causes the same text of the body portion to be repeated in the application document in accordance with the occurrence pattern indicated by the type attribute when the application document is generated from the markup language element ~~wherein mapping includes mapping the properties into a context free chunk element~~; and

store the properties in the markup language document; and

a validation engine configured to validate the markup language document.

19. (Previously presented) The system of Claim 18, wherein the application is further configured to:

determine properties corresponding to an additional mini-document that relates to at least one section of the application document;

map the properties of the additional mini-document into a markup language element, wherein mapping includes mapping the properties into at least one member of a group comprising: a context free chunk element and a table element; and

store the properties of the additional mini-document in the markup language document.

20. (Original) The system of Claim 18, wherein the application is further configured to:

determine whether properties associated with all mini-documents of the application document have been stored in the markup language document; and

process further mini-documents when the properties associated with all mini-documents have not been stored in the markup language document.

21. (Original) The system of Claim 18, wherein the properties of the mini-document stored in the markup language document are understood by an additional application that understands the markup language when the mini-document is not native to the additional application.

22. (Original) The system of Claim 18, wherein the markup language document is manipulated on a server to substantially reproduce the mini-document of the application document notwithstanding the presence of the application that generated the markup language document.